韩山师范学院实验报告

姓名：林婷婷　　专业：信息管理与信息系统　班级：20181171　　学号：2018117126

科目：　Android 应用开发　　实验日期：2021.6.14

实验题目：　后台默默的劳动者，探究 Service

【实验目的】

探究 service

【实验环境及方式】

Android Studio

【实验内容及实验结果】

1. Service 是什么

Service 是 Android 中实现程序后台运行的解决方案，它非常适合执行那些不需要和用户交互而且还要求长期运行的任务。Service 的运行不依赖于任何用户界面，即使程序被切换到后台，或者用户打开了另外一个应用程序，Service 仍然能够保持正常运行。

2. Android 多线程编程、

① 线程的基本用法

Android 多线程编程其实并不比 Java 多线程编程特殊，基本是使用相同的语法。比如，定义一个线程只需要新建一个类继承自 Thread，然后重写父类的 run()方法，并在里面编写耗时逻辑即可。

```
class MyThread : Thread() {
    override fun run() {
        // 编写具体的逻辑
    }
}
```

启动这个线程只需要创建 MyThread 的实例，然后调用它的 start()方法即可，这样 run()方法中的代码就会在子线程当中运行。

```
MyThread().start()
```

② 在子线程中更新 UI

Android 的 UI 是线程不安全的，也就是说，如果想要更新应用程序里的 UI 元素，必须在主线程中进行，否则就会出现异常。对于这种情况，Android 提供了一套异步消息处理机制，完美地解决了在子线程中进行 UI 操作的问题。

```
class MainActivity : AppCompatActivity() {
    val updateText = 1
    val handler = object : Handler() {
        override fun handleMessage(msg: Message) {
            // 在这里可以进行 UI 操作
```

```
                when (msg.what) {
                    updateText -> textView.text = "Nice to meet you"
                }
            }
        }
        override fun onCreate(savedInstanceState: Bundle?) {
            super.onCreate(savedInstanceState)
            setContentView(R.layout.activity_main)
            changeTextBtn.setOnClickListener {
                thread {
                    val msg = Message()
                    msg.what = updateText
                    handler.sendMessage(msg) // 将 Message 对象发送出去
                }
            }
        }
    }
```

3. Service 的基本用法

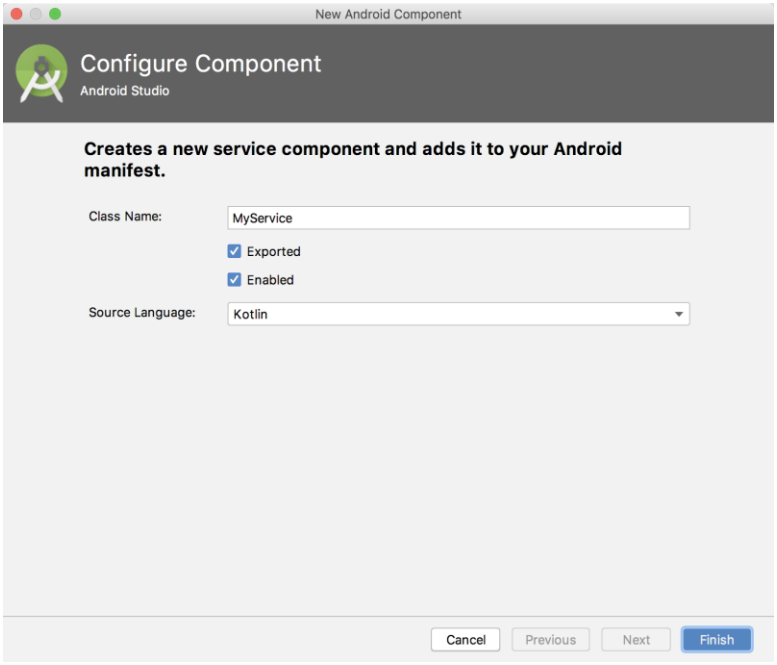
① 定义一个 Service

右键项目的任何包路径→New→Service→Service，会弹出如右图所示的窗口。

Exported 属性表示是否将这个 Service 暴露给外部其他程序访问。

Enabled 属性表示是否启用这个 Service。

点击“ Finish”完成创建。



② 重写 Service

然后需要重写了 onCreate()、onStartCommand()和 onDestroy()这 3 个方法。其中 onCreate()方法会在 Service 创建的时候调用，onStartCommand()方法会在每次 Service 启动的时候调用，onDestroy()方法会在 Service 销毁的时候调用。

③ 在 AndroidManifest 文件中注册

每一个 Service 都需要在 AndroidManifest.xml 文件中进行注册才能生效，这是 Android 四大组件共有的特点。

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.servicetest">
<application …>
<service
    android:name=".MyService"
    android:enabled="true"
    android:exported="true">
</service>
</application>
    </manifest>
```

④ 启动和停止 Service

Context 类中提供了 startService()和 stopService()这两个方法来启动和停止 Service，所以我们在 Activity 里可以直接调用这两个方法。

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        startServiceBtn.setOnClickListener {
            val intent = Intent(this, MyService::class.java)
            startService(intent) // 启动 Service
        }
        stopServiceBtn.setOnClickListener {
            val intent = Intent(this, MyService::class.java)
            stopService(intent) // 停止 Service
        }
    }
}
```

⑤ Activity 和 Service 进行通信

当一个 Activity 和 Service 绑定了之后，就可以调用该 Service 里的 Binder 提供的方法了，如下所示：

```
class MainActivity : AppCompatActivity() {
    lateinit var downloadBinder: MyService.DownloadBinder
    private val connection = object : ServiceConnection {
        override fun onServiceConnected(name: ComponentName, service: IBinder) {
            downloadBinder = service as MyService.DownloadBinder
            downloadBinder.startDownload()
            downloadBinder.getProgress()
        }
        override fun onServiceDisconnected(name: ComponentName) {
        }
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        bindServiceBtn.setOnClickListener {
            val intent = Intent(this, MyService::class.java)
            bindService(intent, connection, Context.BIND_AUTO_CREATE) // 绑定 Service
        }
        unbindServiceBtn.setOnClickListener {
            unbindService(connection) // 解绑 Service
        }
    }
}
```

【教师评语和成绩】

成绩： **指导教师：** **日期：**